gaming machine. Using an EPROM, it was not feasible to store large amounts of game data relating to a complicated 3-D model. Thus, only 2-D object information used to render the 2-D view was stored on the gaming machine.

[0047] However, 2-D games rendered on gaming machines have also become more sophisticated and often employ complex animations. When complicated animations are used in a 2-D system, such as playing movies on a 2-D object, a 3-D system can actually save memory because more types of animation can be used with a 3-D system versus a 2-D system without resorting to using movies, which are memory intensive. In a 2-D system without using movies, the animation properties that may be used are simple two-dimensional movement and color cycling using color palettes, which provide a limited visual appeal.

[0048] When only 2-D information about a 3-D object is available, it is not possible to generate new 2-D views from different viewpoints of the 3-D object. For instance, when a picture of a playing card is rendered on current gaming machines, 3-D information, such as the thickness of the card is not stored. Thus, it is not possible to generate a 2-D view of the playing card from an edge-on viewpoint, because the thickness of the card is not known. As another example, frames from a movie may be used as part of a game presentation on a gaming machine. Each frame of the movie represents a 2-D view from a viewpoint of a camera used to film each frame. If the frame included a picture of a building viewed from the front (e.g., the viewpoint captures the front of the building), it is not possible to generate a new 2-D view of the back of the building using because information regarding the back of the building is not known.

[0049] One advantage of the present invention is the potential game playing area used to present a game of chance modeled in a 3-D gaming environment is greater than the potential game playing area of a 2-D gaming environment. For instance, a game of chance may be presented on each of the six sides of a cube modeled in a virtual gaming environment. To play the game chance, 2-D views of the cube from different viewpoints in the 3-D gaming environment may be rendered in real-time and presented to the player. As described below, in some embodiments, the player may even select the viewpoint in the 3-D gaming environment used to generate the 2-D view.

[0050] On current gaming machines, the cube would be rendered as a 2-D object generated from the 3-D cube as seen from a particular viewpoint. The particular viewpoint is selected when the game is developed and only 2-D information about the cube as viewed from the selected viewpoint would be stored on an EPROM on the gaming machine. Thus, a game of chance could be presented on the sides of the cube rendered from the 2-D object that was generated from the selected viewpoint of the 3-D cube and stored on the EPROM. However, unless additional 2-D objects were generated from different viewpoints, it is not possible to present a game of chance on the sides of the cube not visible from the selected viewpoint because the 2-D object does not store information regarding the sides of the cube not visible from the selected viewpoint. Further, even if multiple 2-D objects were generated, it is difficult and time consuming to generate enough 2-D objects to allow smooth transitions between viewpoints captured by the 2-D objects. It is also difficult to a scale a 2-D object, either smaller or larger, without introducing distortion effects.

[0051] Distortion is also generated when scaling 3-D objects. However, it is easier to deal with using specialized 3-D graphics cards because the card applies a bilinear filtering process to the texels at render time. Without special hardware, such as a 3-D graphics card, it would be difficult to correct for distortion in real-time.

[0052] Finally, in a typical 2-D gaming system, due to the limited flexibility of 2-D, outcomes for a game of chance rendered in 2D and displayed on a gaming machine have to be quantified and pre-rendered i.e. canned animations. Due to the flexibility of a 3-D gaming system the outcomes can be determined through user input giving an unlimited number of animations in response to the players input. By not having to make a series of pre-canned animations but instead determining the animation in response to the players input saves many bytes in storage space requirements. In following figures, details of methods and apparatus used to present a game of chance generated from a 3-D gaming environment are described.

[0053] Returning to **FIG. 1**, the 3-D gaming environment **100** includes three objects: 1) a rectangular box **101** on top of, 2) a plane **114** and **3)** a second box **127**. The box **101**, box **127** and plane **114** are defined in a 3-dimensional rectangular coordinate space **104**. Typically, surfaces of the objects in the gaming environment are defined using a plurality of surface elements. The surface elements may comprise different shapes, such as different types of polygons that are well known in the 3-D graphical arts. For example, the objects in the present information may be defined in a manner to be compatible with one or more graphics standards such as Open Graphics Library (OpenGL). Information on OpenGL may be found at www.opengl.org.

[0054] In one embodiment, the objects in the gaming environment **100** may be defined by a plurality of triangular elements. As an example, a plurality of triangular surface elements **125** are used to define a portion of the surface **108** and the surface face **112**. In another embodiment, the objects in the gaming environment **100**, such as box **101** and box **127**, may be defined by a plurality of rectangular elements. In yet another embodiment, a combination of different types of polygons, such as triangles and rectangles may be used to describe the different objects in the gaming environment **100**. By using an appropriate number of surface elements, such as triangular elements, objects may be made to look round, spherical, tubular or embody any number of combinations of curved surfaces.

[0055] Triangles are by far the most popular polygon used to define 3-D objects because they are the easiest to deal with. In order to represent a solid object, a polygon of at least three sides is required (e.g. triangle). However, OpenGL supports quads, points, lines, triangle strips and quad strips and polygons with any number of points. In addition, 3-D models can be represented by a variety of 3-D curves such as NURBs and Bezier Patches.

[0056] Each of the surface elements comprising the 3-D virtual gaming environment may be described in a rectangular coordinate system or another appropriate coordinate system, such as spherical coordinates or polar coordinates, as dictated by the application. The 3-D virtual gaming environments of the present invention are not limited to the shapes and elements shown in **FIG. 1** or the coordinate system used in **FIG. 1** which are shown for illustrative